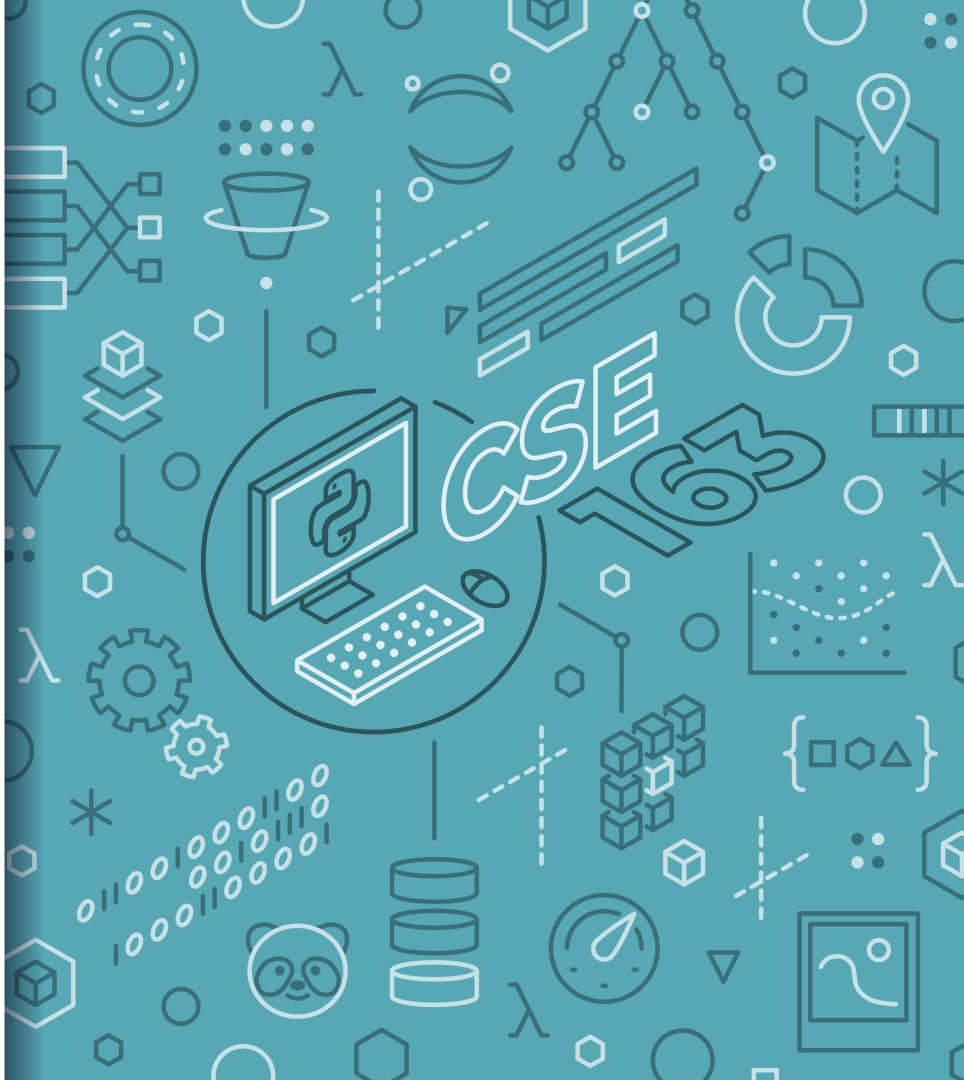
CSE 163

Distributed Computing

Hunter Schafer



Logistics

Exam 2

- Topics
  - Hashing
  - Geo-spatial data
  - Ethics
  - Image Data
- Materials
  - Exam materials posted online
  - Section tomorrow will have another practice exam
  - Lecture Friday will be review

Today

Material that won’t be on the exam, but that you will most likely need to learn if you are working with data in the real world of

Big Data

Programming

So far, we have been writing programs that process data on your computer **sequentially**.

Think of programming like cooking: Follow a recipe to completion

1. Cook the eggs
2. Grate cheese
3. Cut the onions
4. Cook the onions
5. Cut the peppers
6. Cook the peppers
7. Cut fruit
8. Assemble

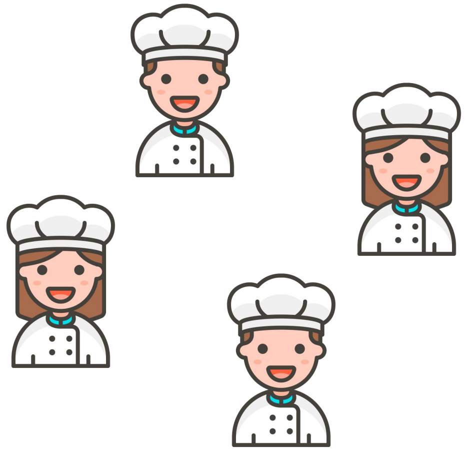
Big Food

This worked great for you for many years until you started having to make the breakfast burritos in bulk.

What do you do? Hire more chefs!

Breakfast Burrito Recipe

1. Cook the eggs
2. Grate cheese
3. Cut the onions
4. Cook the onions
5. Cut the peppers
6. Cook the peppers
7. Cut fruit
8. Assemble



~~Big Food~~
Big Data

Most common way of working with big data is the same:

- Hire more ~~chefs~~ computers

Most of you have dual-core or quad-core computers which means your one machine can do 2-4 things at any point in time

We will not be talking about that sort of parallelism (which is very interesting), but rather talking about distributing work across other computers

Some Sums

Suppose we have a very large dataset of numbers that does not fit in memory and want to compute the sum of the numbers.

This can be solved with a **streaming** algorithm

```
# Not real python
def sum(file):
    f = open(file)
    total = 0
    while f has lines:
        num = f.next_line()
        total += num
    return total
```

This is doable since we just need to store one number in memory

Word Count

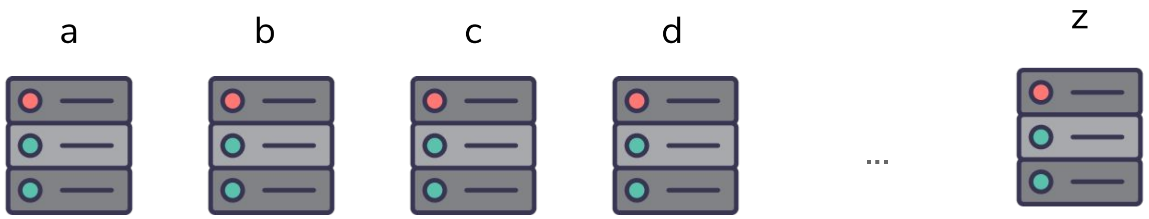
What if instead, I had all of the text in Wikipedia and I wanted to get word counts for each word?

Will streaming work?

- Only if I can store the map of every word in memory
- If I can't, we need to hire more workers to split up the task

How to split up the data?

- What about by starting letter?



Distributed Computing

General Distributed Computing Pattern

- Somehow split up data into separate chunks
- Distribute chunks across your computers so they can each do processing locally
- Either combine the results or have them write the result to a file to be combined later

```
# Not real python
def distribute(data, servers):
    parts = partition(data)
    send(parts, servers)
    results = wait(servers)
    print(results)

def compute(part):
    # do whatever computation with your part
```

Distributed Computing

General Problems in Distributed Computing

- Communication
  - Internet / Networks
- Skew
  - How do you evenly partition the data?
- Reliability
  - Are computers reliable?

Poll Everywhere

Pair

1 minute

pollev.com/cse163

Suppose we have a group of 10,000 computers. The computers we bought are relatively cheap and were advertised to have a mean time to fail of 1 year. Approximately when would we expect the first computer in the group to fail?

Options

- 1 hour
- 12 hours
- 1 day
- 3 days
- 1 week
- 2 weeks
- 1 month
- 3 months
- 6 months
- 9 months
- 1 year

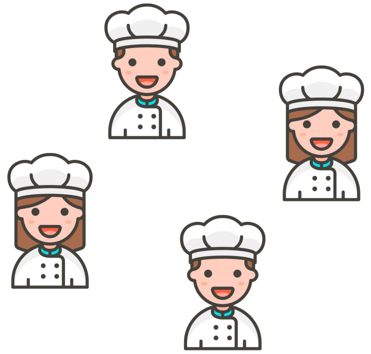
Failures are Common

Computer failures happen all the time when working with large groups of computers

- This would be like one of our chefs disappearing

How to recover from a chef/computer failure?

- Figure out where they went (slow)
- Figure out what they were working on and give that work to someone who isn't busy



Writing code to do this is HARD

Distributed Frameworks

Dealing with the nastiness of distributed computing is not fun

Instead, people use pre-built tools that abstract the details of distributed computation away from the programmer so they can focus on the good stuff

Very popular tools exist that are used across industry

- Apache MapReduce (Google)
- Apache Spark (UC Berkeley / Databricks)

Functional Programming

Functional programming is a slightly different way of thinking about writing code

- In the past we have written code “imperatively”. This means that we have to write all the instructions out (i.e. a loop)
- Wouldn’t it be nice if we could write programs that say “what we want to compute” rather than “how to compute it”?
  - Big idea supported by functional programming

We’ve seen a bit of things that functional programming uses

- Passing functions as parameters

But we are missing some other common things

- Higher order functions

Higher Order Functions

Most languages that support functional programming provide **higher-order functions** to help solve many common tasks.

Commonly, you will have

- Map - transforms each value in a list
- Filter - decides which values to keep in a list
- Reduce - combines the values in a list to one value

Common theme:

- They abstract looping over the values so you just specify the function to compute over the data

MapReduce

The key idea to Apache MapReduce is to apply this to the distributed setting

- You the programmer define a map task and a reduce task
- The software does all the low level details
  - Instead of “looping”, this is distributing work amongst workers

General workflow (**bold** functions are things you write)

- **Map**: Write a function that takes one row of the data and produce key, value pairs to reduce later
- Shuffle the data to group all the values with the same key
- **Reduce:** Write a function that takes a key and a list of values that had that key, and combine them into one value

Let’s go back to the word count example

MapReduce: WordCount

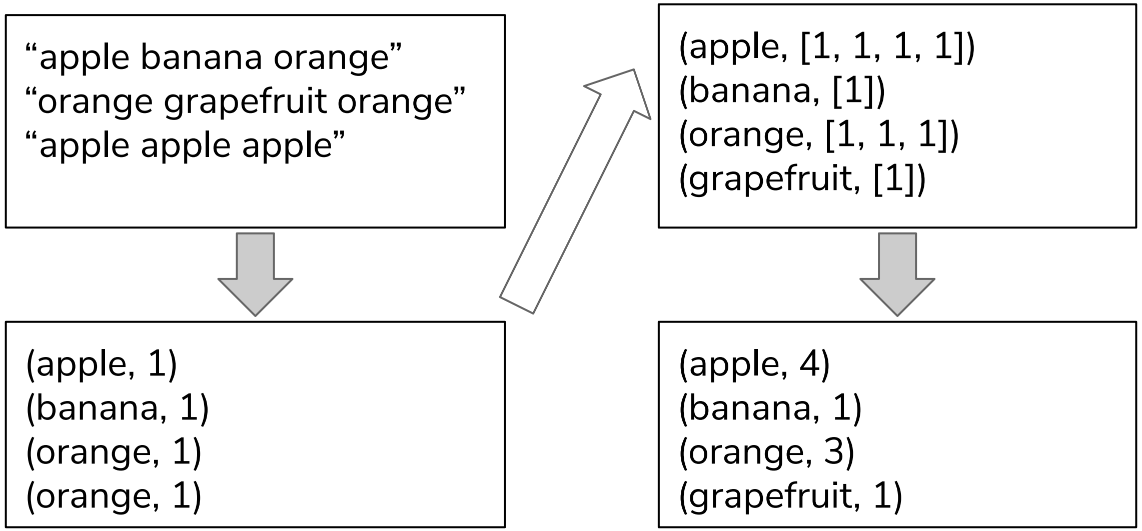
```
def map(document):
    for word in document:
        emitTuple((word, 1))

def reduce(key, values):
    count = 0
    for v in values:
        count += 1
    emitTuple((key, count))
```

MapReduce: WordCount

```
def map(document):
    for word in document:
        emitTuple((word, 1))
```

```
def reduce(key, values):
    count = 0
    for v in values:
        count += 1
    emitTuple((key, count))
```



```
"apple banana orange"
"orange grapefruit orange"
"apple apple apple"
```

```
(apple, 1)
(banana, 1)
(orange, 1)
(orange, 1)
```

```
(apple, [1, 1, 1, 1])
(banana, [1])
(orange, [1, 1, 1])
(grapefruit, [1])
```

```
(apple, 4)
(banana, 1)
(orange, 3)
(grapefruit, 1)
```

MapReduce

MapReduce is relatively simple to write which made it catch on

- A lot of things can be computed in this way
- Programmer doesn't have to worry about distributed at all!

Is everyone using MapReduce?

- Somewhat, but it seems people are moving to Spark
- Google moved away from it for something faster

Why is MapReduce slow?

- All of that tupel writing requires writing things to disk
- Remember, I/O cost is very slow which can slow down the whole process
-

Spark

Spark takes a different approach: it stores everything in memory

This tends to be a lot faster

Spark is also much more flexible since it allows you to write many types of operations (more so than map and reduce)

- This comes at a cost that the API is a bit harder to understand

Spark also comes with an MLlib that has models written to be trained in a distributed fashion